AMENDMENTS TO THE CLAIMS

1. (Currently amended) A method comprising:

saving a definition of a region in a program bounded by an entry breakpoint and an end breakpoint, wherein the entry breakpoint is executed conditionally, wherein a plurality of threads execute the program, wherein the entry breakpoint is executed conditionally further comprises some of the plurality of threads encounter the entry breakpoint and other of the plurality of threads do not encounter the entry breakpoint;

saving a definition of a scoped breakpoint within the region;

if a threadfirst thread of the plurality of threads that executes an instance of the program encounters the entry breakpoint, saving an identifier of the threadfirst thread;

if the threadfirst thread encounters the scoped breakpoint within the region, wherein the region contains the scoped breakpoint and wherein the scoped breakpoint is different from the entry breakpoint and the end breakpoint, determining whether the identifier was saved in response to the threadfirst thread that executes the instance of the program encountering the entry breakpoint;

if the identifier was saved in response to the threadfirst thread that executes the instance of the program encountering the entry breakpoint and the scoped breakpoint within the region was encountered by the threadfirst thread, halting execution of the threadfirst thread that encountered the scoped breakpoint; and

if the identifier was not saved, the threadfirst thread that executes the instance of the program did not encounter the entry breakpoint, and the scoped breakpoint within the region was encountered by the threadfirst thread that executes the instance of the program, allowing execution of the threadfirst thread to continue after the scoped breakpoint was encountered without giving control to a user.

2. (Currently amended) The method of claim 1, further comprising:

after the threadfirst thread encounters the end breakpoint, removing the identifier of the threadfirst thread that was saved.

3. (Currently amended) The method of claim 2, further comprising:

3

Docket No. ROC920030386US1
Serial No. 10/821,148

allowing execution of the ~~thread~~first thread to continue upon the ~~thread~~first thread encountering the end breakpoint without giving control to the user.

4. (Currently amended) The method of claim 3, further comprising:

allowing execution of the ~~thread~~first thread to continue upon the ~~thread~~first thread encountering the entry breakpoint without giving control to the user.

5. (Currently amended) An apparatus comprising:

means for saving a definition of a region in a program bounded by an entry breakpoint and an end breakpoint, wherein the entry breakpoint is executed conditionally, wherein a plurality of threads execute the program, wherein the entry breakpoint is executed conditionally further comprises some of the plurality of threads encounter the entry breakpoint and other of the plurality of threads do not encounter the entry breakpoint;

means for saving a definition of a scoped breakpoint within the region;

means for saving an identifier of a ~~thread~~first thread of the plurality of threads that executes an instance of the program if the ~~thread~~first thread that executes the instance of the program encounters the entry breakpoint;

means for determining whether the identifier was saved in response to the ~~thread~~first thread that executes the instance of the program encountering the entry breakpoint if the ~~thread~~first thread encounters the scoped breakpoint within the region, wherein the region contains the scoped breakpoint and wherein the scoped breakpoint is different from the entry breakpoint and the end breakpoint;

means for halting execution of the ~~thread~~first thread that encountered the scoped breakpoint if the identifier was saved in response to the ~~thread~~first thread that executes the instance of the program encountering the entry breakpoint and the scoped breakpoint within the region was encountered by the ~~thread~~first thread that executes the instance of the program; and

means for allowing execution of the ~~thread~~first thread to continue after the scoped breakpoint was encountered without giving control to a user if the identifier was not

4   Docket No. ROC920030386US1
                              Serial No. 10/821,148

saved, the ~~thread~~first thread that executes the instance of the program did not encounter the entry breakpoint, and the scoped breakpoint <u>within the region</u> was encountered by the ~~thread~~first thread that executes the instance of the program.

6. (Currently amended) The apparatus of claim 5, further comprising:

means for removing the identifier of the ~~thread~~first thread that was saved after the ~~thread~~first thread encounters the end breakpoint.

7. (Currently amended) The apparatus of claim 6, further comprising:

means for allowing execution of the ~~thread~~first thread to continue upon the ~~thread~~first thread encountering the end breakpoint without giving control to the user.

8. (Currently amended) The apparatus of claim 7, further comprising:

means for allowing execution of the ~~thread~~first thread to continue upon the ~~thread~~first thread encountering the entry breakpoint without giving control to the user.

9. (Currently amended) A <u>computer-readable</u> storage medium encoded with instructions, wherein the instructions when executed comprise:

saving a definition of a region in a program bounded by an entry breakpoint and an end breakpoint, wherein the entry breakpoint is executed conditionally<u>, wherein a plurality of threads execute the program, wherein the entry breakpoint is executed conditionally further comprises some of the plurality of threads encounter the entry breakpoint and other of the plurality of threads do not encounter the entry breakpoint</u>;

saving a definition of a scoped breakpoint within the region;

if a ~~thread~~<u>first thread of the plurality of threads</u> that executes an instance of the program encounters the entry breakpoint, saving an identifier of the ~~thread~~first thread;

if the ~~thread~~first thread encounters the scoped breakpoint within the region, <u>wherein the region contains the scoped breakpoint and wherein the scoped breakpoint is different from the entry breakpoint and the end breakpoint,</u> determining whether the

5

identifier was saved in response to the ~~thread~~first thread that executes the instance of the program encountering the entry breakpoint;

if the identifier was saved in response to the ~~thread~~first thread that executes the instance of the program encountering the entry breakpoint and the scoped breakpoint within the region was encountered by the ~~thread~~first thread that executes the instance of the program, halting execution of the ~~thread~~first thread that encountered the scoped breakpoint; and

if the identifier was not saved, the ~~thread~~first thread that executes the instance of the program did not encounter the entry breakpoint, and the scoped breakpoint within the region was encountered by the ~~thread~~first thread that executes the instance of the program, allowing execution of the ~~thread~~first thread to continue after the scoped breakpoint was encountered without giving control to a user.

10. (Currently amended) The computer-readable storage medium of claim 9, further comprising:

allowing execution of the ~~thread~~first thread to continue upon the ~~thread~~first thread encountering the entry breakpoint without giving control to the user.

11. (Currently amended) The computer-readable storage medium of claim 10, further comprising:

allowing execution of the ~~thread~~first thread to continue upon the ~~thread~~first thread encountering the end breakpoint without giving control to the user.

12. (Currently amended) The computer-readable storage medium of claim 11, further comprising:

after the ~~thread~~first thread encounters the end breakpoint, removing the identifier of the ~~thread~~first thread that was saved.

13. (Currently amended) A computer system comprising:

a processor; and

6

Docket No. ROC920030386US1
Serial No. 10/821,148

memory encoded with instructions, wherein the instructions when executed on the processor comprise:

saving a definition of a region in a program bounded by an entry breakpoint and an end breakpoint, wherein the entry breakpoint is executed conditionally, wherein a plurality of threads execute the program, wherein the entry breakpoint is executed conditionally further comprises some of the plurality of threads encounter the entry breakpoint and other of the plurality of threads do not encounter the entry breakpoint,

saving a definition of a scoped breakpoint within the region,

if a ~~thread~~first thread of the plurality of threads that executes an instance of the program encounters the entry breakpoint, saving an identifier of the ~~thread~~first thread,

if the ~~thread~~first thread encounters the scoped breakpoint within the region, wherein the region contains the scoped breakpoint and wherein the scoped breakpoint is different from the entry breakpoint and the end breakpoint, determining whether the identifier was saved in response to the ~~thread~~first thread that executes the instance of the program encountering the entry breakpoint,

if the identifier was saved in response to the ~~thread~~first thread that executes the instance of the program encountering the entry breakpoint and the scoped breakpoint within the region was encountered by the ~~thread~~first thread, halting execution of the ~~thread~~first thread that encountered the scoped breakpoint, and

if the identifier was not saved, the ~~thread~~first thread that executes the instance of the program did not encounter the entry breakpoint, and the scoped breakpoint within the region was encountered by the ~~thread~~first thread that executes the instance of the program, allowing execution of the ~~thread~~first thread to continue after the scoped breakpoint was encountered without giving control to a user.

7

Docket No. ROC920030386US1
Serial No. 10/821,148

14. (Currently amended) The computer system of claim 13, wherein the instructions further comprise:

after the ~~thread~~first thread encounters the end breakpoint, removing the identifier of the ~~thread~~first thread that was saved.

15. (Currently amended) The computer system of claim 14, wherein the instructions further comprise:

allowing execution of the ~~thread~~first thread to continue upon the ~~thread~~first thread encountering the entry breakpoint without giving control to the user.

16. (Currently amended) The computer system of claim 15, wherein the instructions further comprise:

allowing execution of the ~~thread~~first thread to continue upon the ~~thread~~first thread encountering the end breakpoint without giving control to the user.

17. (Currently amended) A method of configuring a computer, comprising:

configuring the computer to save a definition of a region in a program bounded by an entry breakpoint and an end breakpoint, wherein the entry breakpoint is executed conditionally, wherein a plurality of threads execute the program, wherein the entry breakpoint is executed conditionally further comprises some of the plurality of threads encounter the entry breakpoint and other of the plurality of threads do not encounter the entry breakpoint;

configuring the computer to save a definition of a scoped breakpoint within the region;

configuring the computer to save an identifier of a ~~thread~~first thread of the plurality of threads that executes an instance of the program if the ~~thread~~first thread that executes the instance of the program encounters the entry breakpoint;

configuring the computer to determine whether the identifier was saved in response to the ~~thread~~first thread that executes the instance of the program encountering the entry breakpoint if the ~~thread~~first thread encounters the scoped breakpoint within the

Docket No. ROC920030386US1
Serial No. 10/821,148

region, wherein the region contains the scoped breakpoint and wherein the scoped
breakpoint is different from the entry breakpoint and the end breakpoint.;

configuring the computer to halt execution of the ~~thread~~first thread that
encounters the scoped breakpoint within the region if the identifier was saved in response
to the ~~thread~~first thread that executes the instance of the program encountering the entry
breakpoint and the scoped breakpoint was encountered by the ~~thread~~first thread that
executes the instance of the program; and

configuring the computer to allow execution of the ~~thread~~first thread to continue
after the scoped breakpoint was encountered without giving control to a user if the
identifier was not saved, the ~~thread~~first thread that executes the instance of the program
did not encounter the entry breakpoint, and the scoped breakpoint within the region was
encountered by the ~~thread~~first thread that executes the instance of the program.

18. (Currently amended) The method of claim 17, further comprising:

configuring the computer to remove the saved identifier of the ~~thread~~first thread
after the ~~thread~~first thread encounters the end breakpoint.

19. (Currently amended) The method of claim 18, further comprising:

configuring the computer to allow execution of the ~~thread~~first thread to continue
upon the ~~thread~~first thread encountering the end breakpoint without giving control to the
user.

20. (Currently amended) The method of claim 19, further comprising:

configuring the computer to allow execution of the ~~thread~~first thread to continue
upon the ~~thread~~first thread encountering the entry breakpoint without giving control to
the user.

9   Docket No. ROC920030386US1
                              Serial No. 10/821,148